

REMARKS

Claims 1 to 28 are pending in this application. Of these, claims 1 and 14 are independent.¹ Favorable reconsideration and further examination are respectfully requested.

Initially, claim 19 was objected to for an improper dependency. As shown above, claim 19 has been corrected. Withdrawal of the objection thereto is respectfully requested.

The claims were rejected over U.S. Patent Publication No. 2002/0168621 (Cook). As shown above, Applicants have amended the claims to define the invention with greater clarity. These amendments, however, were not made in response to the rejection over Cook and, therefore, this should be viewed as a traversal of the rejection over Cook.

Amended independent claim 1 recites a method, performed by one or more processing devices, for use in an electronic learning system that stores information as learning objects. The method comprises designating a target learning object as a project object, and storing version dependency data in the project object. The version dependency data identifies at least a version of a first object upon which the project object directly depends, and a version of a second object upon which the project object indirectly depends.

Cook is not understood to disclose or to suggest the foregoing features of claim 1.

Referring to Fig. 1, Cook describes an system that provides a

an agent (108) for each student (101) which adapts to its student, and provides individualized guidance to the student and controls to the augmented computer assisted instructional materials. The instructional materials of this invention are augmented to communicate the student's performance and the material's pedagogical characteristics to the agent, and to receive control from the agent...the agent maintains data reflecting the student's pedagogic or cognitive characteristics in a protected and portable media in the personal control of the student. (Abstract)

¹ The Examiner is urged to independently confirm this recitation of the pending claims.

The agent monitors a student's instruction, and builds a student data object 109, which is unique to each student. As described on page 8:

As depicted by arrow 110, the student data object is referenced by the agent in order to generate its actions and is updated by the agent as it processes events and student meta-requests. (00091)

The student data objects may contain student data 116, which may be accessed by school teachers and administrators (0092 and 0093).

Fig. 2 shows a client system, which may be used by a student defined by an object.

According to Cook,

In cases of low-bandwidth network access, the client system can have one or more disc drives 209 which can be used as a pre-fetch buffer or a read-only cache. The disks preferably are magnetic, although in a less preferable embodiment they can also include CDROMs. This optional capability serves to enhance communications efficiency in cases where the network has relatively low bandwidth. Large files can be downloaded in advance of a student session or the student client can cache read-only data across sessions obviating the need for downloading such files. Such caching requires the operating system components *to maintain some form of version control of the read-only data*. (0097) (emphasis added)

Thus, Cook does mention some form of version control. Furthermore, as described on page 10 of Cook,

In the preferred embodiment using portable media, the executive software verifies that the contained student data object belongs to the identified student and accesses the methods of the student data object upon request, primarily, from agent software 225. In an embodiment with centrally stored student data objects, the executive software downloads the identified student's data object from the student database... The student model is updated by the agent during the student session and modified parts are stored back in portable media 240 or uploaded to the student database on a server. (0104)

Thus, Cook also describes downloading a student data object to portable media, and uploading a student data object from portable media to a database on a server.

Thus, while Cook does describe moving data objects between locations (portable media and server), and does mention version control briefly and generally, there is nothing in Cook to disclose or to suggest the features of claim 1, in particular, storing version dependency data in the project object, where the version dependency data identifies at least a version of a first object upon which the project object directly depends, and a version of a second object upon which the project object indirectly depends.

The Office Action cites paragraphs 0077 and 0079 through 0081 for their alleged disclosure of the foregoing features. Applicants respectfully disagree. Applicants note that the Office Action merely restates the claim language and then cites to the above paragraphs, without any accompanying explanation. As such, Applicants are forced to guess at which features of Cook are being cited to correspond to the features of claim 1.

In this regard, Applicants acknowledge the existence of student data object 109, which is described above. Since student data object 109 is the only data object referenced in paragraph 0077 (cited for its alleged disclosure of a project object), Applicants assume that the Office Action is equating that to the claim's project object. However, there is no indication whatsoever that student data object 109 stores version dependency data identifying objects upon which student data object 109 depends directly and upon which student data object 109 depends indirectly. Fig. 1 does show arrows going to and from student data object 109. However, these arrows do not correlate to stored dependency data, but rather communication between student

data object 109 and the other non-object elements of Fig. 1. For example arrow 113 is referring to teachers and administrators 106, who are clearly not data objects. Arrow 110 is referring to agent software 108, which is also not a data object, but rather software used to control a student's instruction and to build the student data object (see 0083).

Even if one were to assume (incorrectly) that the arrows of Fig. 1 constitute dependencies, and that agent software 108 is a data object, at the very best Fig. 1 would show only direct dependencies to student data object 109. Claim 1 requires that the project object store dependency data that identifies an indirect dependency, which is clearly not shown in Fig. 1, even under the most strained interpretation of Cook provided in this paragraph.

Paragraphs 0079 through 0081 describe teachers authoring objects that contain educational materials to be presented to students, as follows:

Also, instructional designers can create, or "author," materials for use in this invention. Such materials can be original or can be derived from existing textbooks, workbooks or media material. They can employ standardized curricula, pretests such as criterion tests, post-tests, and standardized texts. Authoring instructional materials in a course suitable for interactive instruction typically comprises several steps, including decisions about the objects to display to the student, the sequencing of these objects, and the interactions with the agent. The first step is the selection of objects which carry the education content for presentation to a student. Objects can include visual display items, such as text, graphics, animation or movies, audible display items, such as voice, audio and so forth...The second step is the selection of the sequencing logic for the ordered display of the objects to the student and the educationally appropriate reaction to student requests and responses. (0081)

It is possible that the Office Action is equating these educational objects of paragraph 0081 to the first and second data objects of claim 1. However, student data object 109 (the supposed counterpart to the claim's project object) does not store dependency data for these educational objects and, therefore, they cannot be equated to the first and second data objects of claim 1.

For at least the foregoing reasons, Cook is not believed to disclose or to suggest at least storing version dependency data in the project object, where the version dependency data identifies at least a version of a first object upon which the project object directly depends, and a version of a second object upon which the project object indirectly depends. Accordingly, claim 1 is believed to be patentable over Cook. Independent claim 14 is a program claim that roughly corresponds to claim 1, and is also believed to be patentable.

Throughout the Office Action, the Examiner simply repeats the claim language, and cites paragraphs without any explanation. As best understood by Applicants, the closest passages to the claimed versioning system are those general statements referred to above regarding versioning. These passages, however, are not understood to disclose or to suggest features of the dependent claims, such as updating version dependency data (claim 10) and resolving dependencies (claim 13).

In this regard, Applicants have also added new dependent claims 27 and 28 into the application. According to these claims, the version of the first object and the version of the second object store object dependency data but not version dependency data, the object dependency data for the version of the first object identifies one or more first learning objects upon which the version of the first object depends but does not identify versions of the one or more first learning objects, and object dependency data for the version of the second object identifies one or more second learning objects upon which the version of the second object depends but does not identify versions of the one or more second learning objects. There is nothing whatsoever in Cook that discloses or suggests a distinction between object dependency

data and version dependency data, as claims. Accordingly, these claims are believed to be patentable over Cook.

The remaining dependent claims are also believed to define patentable features of the invention. Each dependent claim partakes of the novelty of its corresponding independent claim and, as such, each has not been discussed specifically herein.

It is believed that all of the pending claims have been addressed. However, the absence of a reply to a specific rejection, issue or comment does not signify agreement with or concession of that rejection, issue or comment. In addition, because the arguments made above may not be exhaustive, there may be reasons for patentability of any or all pending claims (or other claims) that have not been expressed. Finally, nothing in this paper should be construed as an intent to concede any issue with regard to any claim, except as specifically stated in this paper, and the amendment of any claim does not necessarily signify concession of unpatentability of the claim prior to its amendment.

In view of the foregoing amendments and remarks, Applicants respectfully submit that the application is in condition for allowance, and such action is respectfully requested at the Examiner's earliest convenience.

Applicants' undersigned attorney can be reached at the address shown below. All telephone calls should be directed to the undersigned at 617-521-7896.

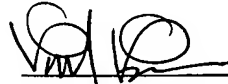
Applicants: Wolfgang Theilmann, et al.
Serial No. : 10/809,873
Filed : March 25, 2004
Page : 16 of 16

Attorney's Docket No.: 13909-161001
Client Ref.: 2004P00116US

Please apply any fees or credits due in this case, which are not already covered by check,
to Deposit Account 06-1050 referencing Attorney Docket No. 13909-161001.

Respectfully submitted,

Date: December 5, 2006



Paul A. Pysher
Reg. No. 40,780

Fish & Richardson P.C.
225 Franklin Street
Boston, MA 02110-2804
Telephone: (617) 542-5070
Facsimile: (617) 542-8906

21497623.doc